Assignment Report

Performance Stats.

Sayed Kotb Sayed

28th September, 2018

Selection Sort

```
1000_unsorted: 1.76032                           5000_unsorted: 40.2838
1000_sorted: 1.64569                             5000_sorted: 37.9107

10000_unsorted: 166.444                          50000_unsorted: 3881.14
10000_sorted: 152.245                            50000_sorted: 3856.17

75000_unsorted: 8873.48                          100000_unsorted: 15541.9
75000_sorted: 8678.19                            100000_sorted: 15488.9

500000_unsorted: 400517                          500000_sorted: 395923
```

Comments:
1- We can notice the quadratic growth in the execution time.
2- We can also notice the data independence as the time take to sort the unsorted or the sorted datasets is almost equal.

Insertion Sort

```
1000_unsorted: 1.02113                           5000_unsorted: 24.5629
1000_sorted: 0.006934                            5000_sorted: 0.033244

10000_unsorted: 89.3198                          50000_unsorted: 1962.56
10000_sorted: 0.065214                           50000_sorted: 0.234192

75000_unsorted: 4449.39                          100000_unsorted: 7903.48
75000_sorted: 0.320534                           100000_sorted: 0.424844

500000_unsorted: 211913                          500000_sorted: 2.14973
```

Comments:
1- We can notice the quadratic growth in the execution time.
2- We can also notice the data dependence as the time take to sort the sorted dataset is much less than the time taken to sort the unsorted dataset, almost constant (Best case).

Merge Sort

```
1000_unsorted: 0.797598                           5000_unsorted: 3.88383
1000_sorted: 0.466235                             5000_sorted: 2.51888

10000_unsorted: 8.17525                           50000_unsorted: 38.2267
10000_sorted: 7.41998                             50000_sorted: 25.8856

75000_unsorted: 55.2401                           100000_unsorted: 73.5054
75000_sorted: 38.8808                             100000_sorted: 52.0414

500000_unsorted: 366.693                          500000_sorted: 282.72
```

Comments:
1- We can notice the n.log(n) growth in the execution time.
2- We can also notice the data independence as the time take to sort the unsorted or the sorted datasets is almost equal.

Quick Sort

```
1000_unsorted: 0.249443                          5000_unsorted: 1.15269
1000_sorted: 7.16056                             5000_sorted: 180.411

10000_unsorted: 2.44816                          50000_unsorted: 14.8086
10000_sorted: 678.933                            50000_sorted: 16773.5

75000_unsorted: 26.7181                          100000_unsorted: 32.3334
75000_sorted: 38290.1                            100000_sorted: 71294.8

500000_unsorted: 168.319                    500000_unsorted: Stackoverflow
```

Comments:
1- We can notice the n.log(n) growth in the execution time.
2- We can also notice the data dependence as the time take to sort the sorted dataset is much higher than the time taken to sort the unsorted dataset, almost quadratic (Worst case).
3- The largest dataset overflows the stack because it hits the worst case.

Heap Sort

```
1000_unsorted: 0.45886                              5000_unsorted: 1.8786
1000_sorted: 0.320742                               5000_sorted: 1.65086

10000_unsorted: 4.07218                             50000_unsorted: 23.2341
10000_sorted: 3.68997                               50000_sorted: 27.5959

75000_unsorted: 39.3381                             100000_unsorted: 50.0789
75000_sorted: 31.4483                               100000_sorted: 42.8571

500000_unsorted: 320.093                            500000_sorted: 252.768
```

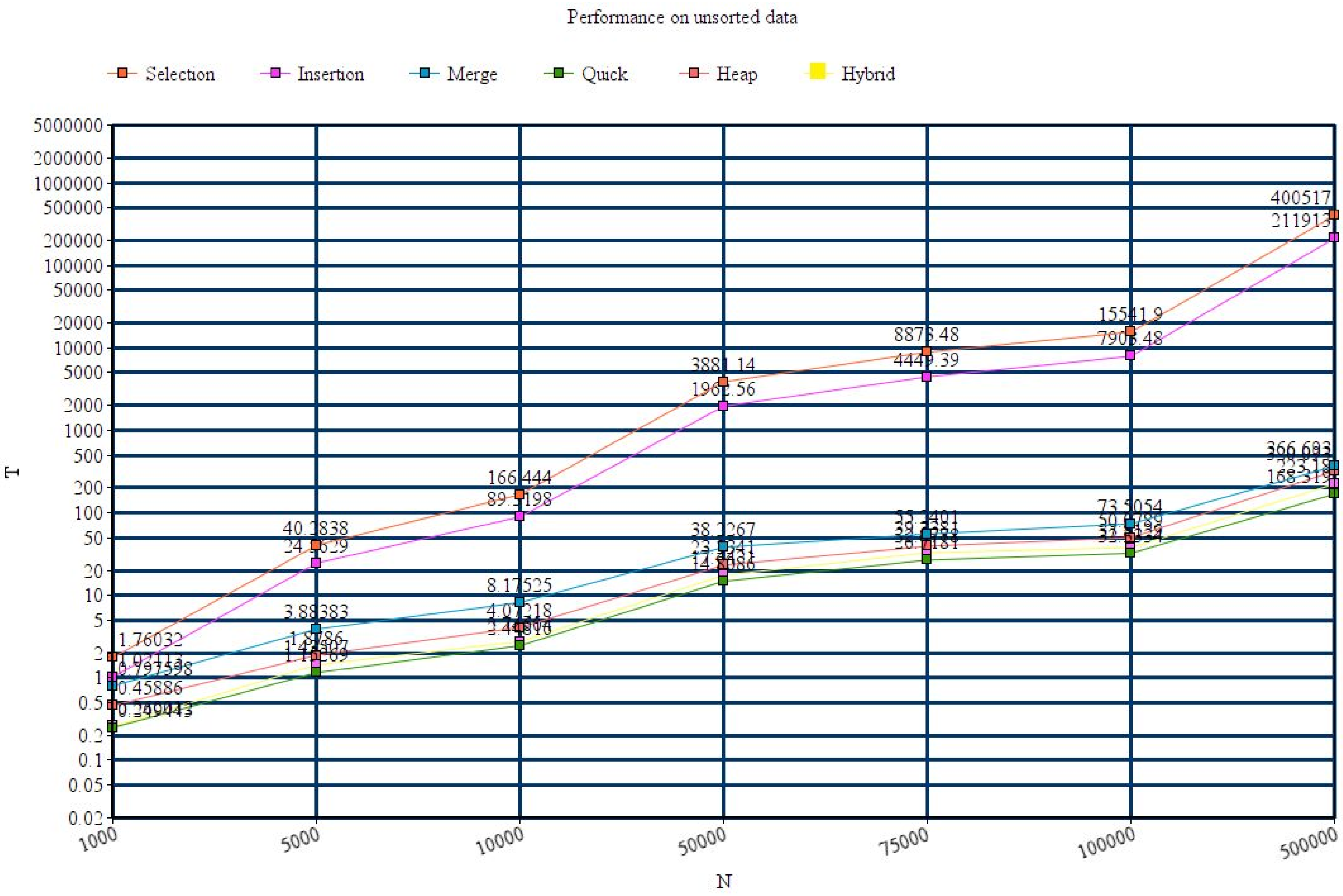
Comments:
1- We can notice the n.log(n) growth in the execution time.
2- We can also notice the data independence as the time take to sort the unsorted or the sorted datasets is almost equal.

Hybrid Sort (TimSort)

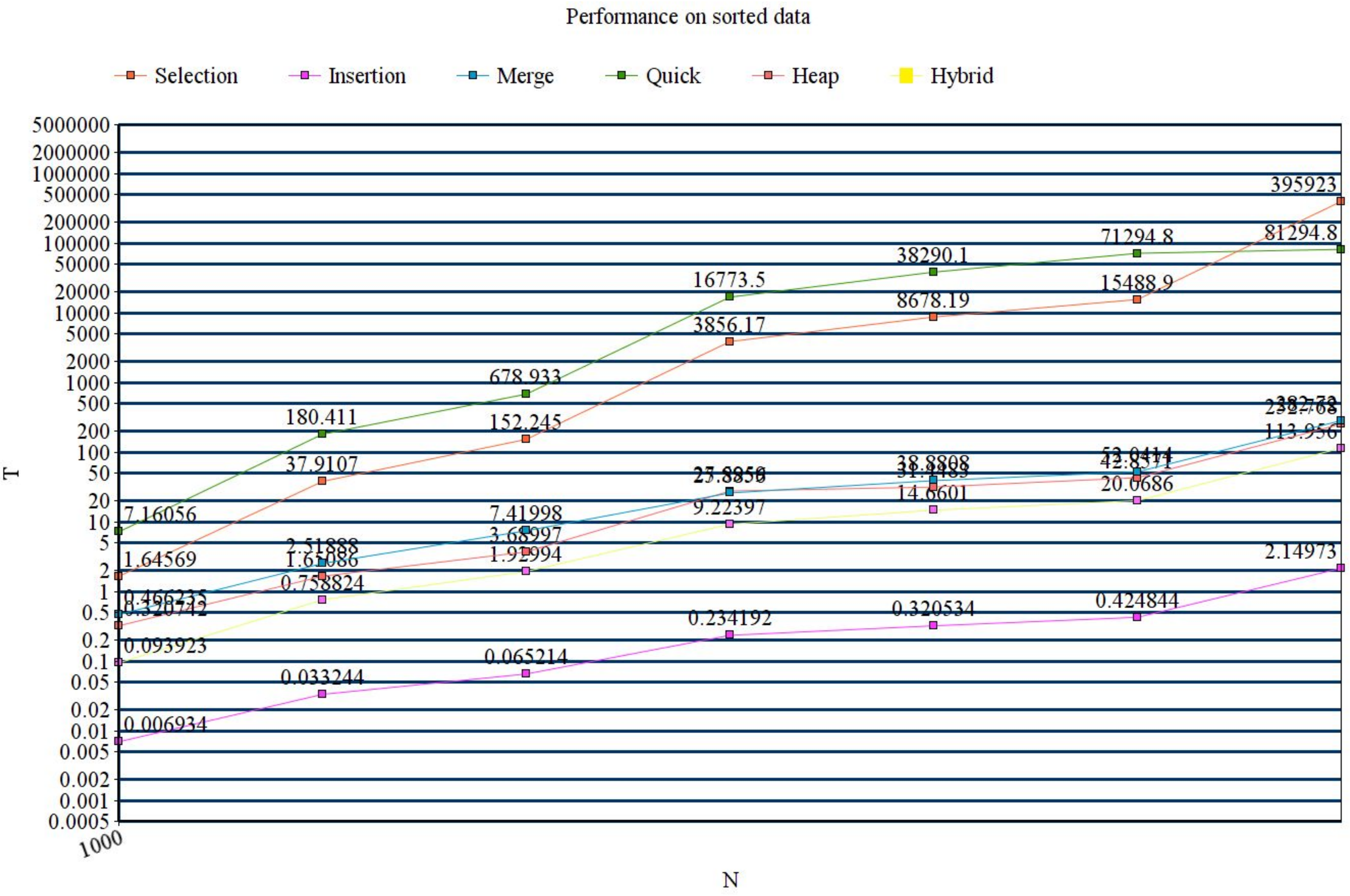
```
1000_unsorted: 0.260012                         5000_unsorted: 1.43507
1000_sorted: 0.093923                           5000_sorted: 0.758824

10000_unsorted: 2.74604                         50000_unsorted: 17.4421
10000_sorted: 1.92994                           50000_sorted: 9.22397

75000_unsorted: 32.8778                         100000_unsorted: 37.8129
75000_sorted: 14.6601                           100000_sorted: 20.0686

500000_unsorted: 223.18                         500000_sorted: 113.956
```

Comments:
1- We can notice the n.log(n) growth in the execution time.
2- We can also notice the data independence as the time take to sort the unsorted or the sorted datasets is almost equal.
3- We can notice about 40% improvement in the runtime compared to the conventional Merge Sort.

Plot (Unsorted)



Plot (Sorted)


Performance on sorted data
Selection
Insertion
Merge
Quick
Heap
Hybrid
T
N
1000
5000000
2000000
1000000
500000
200000
100000
50000
20000
10000
5000
2000
1000
500
200
100
50
20
10
5
2
1
0.5
0.2
0.1
0.05
0.02
0.01
0.005
0.002
0.001
0.0005
395923
81294.8
71294.8
38290.1
16773.5
15488.9
8678.19
3856.17
678.933
180.411
152.245
37.9107
7.16056
1.64569
0.466235
0.320742
0.093923
0.006934
2.51888
1.65086
0.758824
0.033244
7.41998
3.68997
1.92994
0.065214
9.22397
0.234192
38.8808
14.6601
0.320534
52.0414
42.8571
20.0686
0.424844
113.956
2.14973